ИНФОРМАТИКА, ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА И УПРАВЛЕНИЕ
INFORMATION TECHNOLOGY, COMPUTER SCIENCE, AND MANAGEMENT



УДК 681.3.06 https://doi.org/10.23947/2687-1653-2020-20-4-422-429

Полиномиально вычислимые Σ-спецификации иерархизированных моделей реагирующих систем



В. Н. Глушкова, К. С. Коровина
ФГБОУ ВО «Донской государственный технический университет» (г. Ростов-на-Дону, Российская Федерация)

Введение. Пакеты верификации проектируют и анализируют корректность параллельных и распределенных систем в рамках различных классов темпоральных логик линейного и ветвящегося времени. В статье рассматривается полиномиально реализуемый класс $\Delta_0 T$-формул, интерпретируемый на многосортных моделях с иерархическими надстройками. Структура надстройки описывается произвольной контекстно-свободной (КС) грамматикой. Предикаты и функции сигнатуры модели интерпретируются на исходном КС-списке, достраиваемом в процессе интерпретации.
Материалы и методы. Для теорий из $\Delta_0 T$-квазитождеств, обладающих свойствами нётеровости и конфлюентности, строится константная модель. Рассматриваются формулы многосортного языка исчисления предикатов (ИП) 1-го порядка с переменными сорта «список», интерпретируемые на моделях с иерархизированной надстройкой. Теория интерпретируется на деревьях вывода грамматики, описывающих поведение специфицируемой системы. Правила КС-грамматики иерархизируют пространство действий моделируемой системы. Отмечено, что для моделирования систем реального времени недостаточно выразительных возможностей $\Delta_0 T$-формул. Поэтому для спецификации используются выражения с неограниченным квантором всеобщности ∀, известные как ПТ-формулы.
Результаты исследования. В качестве примера приводится логическая спецификация автоматизированного комплекса, который состоит из манипулятора, обрабатывающего детали. Положение позиций фиксируется датчиками. Описывается цикл работы манипулятора. Спецификация его функционирования состоит в иерархизации действий правилами КС-грамматики и их описании формулами ИП 1-го порядка с учетом значений времени.
Обсуждение и заключения. В статье показано, что класс рассмотренных формул можно использовать для моделирования систем реального времени. Приводится пример логической спецификации управляющего устройства поведением манипулятора.

Ключевые слова: логическая спецификация, модель теории, реагирующая система, КС-грамматика, формула ИП 1-го порядка.

Для цитирования: Глушкова, В. Н. Полиномиально вычислимые Σ-спецификации иерархизированных моделей реагирующих систем / В. Н. Глушкова, К. С. Коровина // Advanced Engineering Research. — 2020. — Т. 20, № 4. — С. 422–429. https://doi.org/10.23947/2687-1653-2020-20-4-422-429



Polynomially computable Σ–specifications of hierarchical models of reacting systems

V. N. Glushkova, K. S. Korovina
Don State Technical University (Rostov-on-Don, Russian Federation)

Introduction. Verification packages design and analyze the correctness of parallel and distributed systems within the framework of various classes of temporal logics of linear and branching time. The paper discusses a polynomially realizable class of $\Delta_0 T$ -formulas interpreted on multi-sorted models with hierarchical suspensions. The suspension

structure is described by an arbitrary context-free (CF) grammar. The predicates and functions of the model signature are interpreted on the original CF-list, which is completed during the interpretation process.
Materials and Methods. A constant model is constructed for theories from $\Delta_0 T$-quasiidentities with Noetherian and confluence properties. We consider formulas of the multi-sorted first-order predicate calculus (PC) language with variables of the “list” sort interpreted on models with a hierarchized suspension. The theory is interpreted in terms of grammar inference trees describing the behavior of the specified system. The CF-grammar rules hierarchize the action space of the modeled system. It is noted that the expressive capabilities of $\Delta_0 T$-formulas are insufficient for modeling real-time systems. Therefore, expressions with unbounded universal quantifier ∀, known as PT formulas, are used for the specification.
Results. The logical specification of an automated complex which consists of a workpiece manipulator is given as an example. The location of the positions is fixed by sensors. The operating cycle of the manipulator is described. The specification of its operation consists in the hierarchization of actions by the rules of the CF-grammar and their description by the first-order PT-formulas taking into account the time values.
Discussion and Conclusions. The paper shows that the class of the considered formulas can be used to model real-time systems. An example of the logical specification of a manipulator behavior control device is given.

***Keywords**:* logical specification, theory model, reactive system, CF-grammar, first-order PC-formula.

For citation: V. N. Glushkova, K. S. Korovina. Polynomially computable Σ–specifications of hierarchical models of reacting systems. Advanced Engineering Research, 2020, vol. 20, no. 4, p. 422–429. https://doi.org/10.23947/2687-1653-2020-20-4-422-429

Введение. Математически обоснованные практически значимые методы верификации сложных программных и технических систем базируются на аппарате математической логики [1–4]. Известна методика применения этого подхода для различных видов реагирующих систем реального времени (протоколов связи, систем управления, встроенных бортовых систем космической техники и др.). Данная методика позволяет верифицировать системы model checking [5–7]. Многочисленные пакеты верификации поддерживают проектирование и анализ корректности параллельных и распределенных систем в рамках различных классов темпоральных логик линейного и ветвящегося времени: LTL, CTL, TCTL и др. [8].

Для моделирования времени в этих системах используется стандартная модель временного автомата. Это конечный автомат, снабженный переменными специального вида — локальными часами. Количественный анализ временных характеристик системы затруднен сложными экспоненциальными алгоритмами построения часовых поясов как классов эквивалентности [9]. Поэтому для упрощения анализа необходимо разработать более выразительный практически значимый языка спецификации.

Предлагается использовать для моделирования язык Σ-спецификаций, выделенный в концепции семантического программирования, которая основана на теоретико-модельном подходе [10]. В этом случае для построения формальной модели анализируемой системы можно использовать язык исчисления предикатов 1-го порядка, расширенный аксиомами для операций над списочными структурами[1,2,3,4].

Материалы и методы. В работе используется терминология статей [11, 12]. Пусть $\mathcal{M}$ — многосортная модель сигнатуры $\sigma = < I, C, F, R>$. Здесь I — множество сортов, включая сорт «список» (*list*). C, F, R — множества констант, функций и предикатов соответственно. Все символы сигнатуры имеют тип. Если $f \in F$ является n-местной функцией, $n \geq 0$, то ее тип равен $<i_1, \dots i_n, i>$, где $i_1, \dots i_n$, $i \in I$, причем $i_1, \dots i_n$ — типы аргументов, i — тип значения функции. Аналогично n-местный предикат $r \in R$ имеет тип $< i_1, \dots i_n >$. Носитель модели $\mathcal{M}$ — индексированное семейство множеств $U_j = C_j$, $j \in I$, где C_j — множество констант сорта j; f:

$$U_{i_1} \times \ldots \times U_{i_n} \to U_i \, , \, r \subseteq U_{i_1} \times \ldots \times U_{i_n} \, .$$

Для модели $\mathcal{M}$ над множеством констант C формируется списочная надстройка $D_G(C)$ из иерархизированных КС-списков, структура которых задается КС-грамматикой $G = (N, T, P)$. Здесь N, T — множества нетерминальных и терминальных символов. Множество $D_G(C)$ определяется как наименьшее множество всех

[1] Глушкова В. Н. Верификация робототехнических иерархических систем реального времени // Математическое моделирование и биомеханика в современном университете : тез. X Всерос. шк.-семинара. Ростов н/Д ; Таганрог : ЮФУ, 2015. С. 32.
[2] Глушкова В. Н. Σ-спецификация робототехнических систем реального времени // Алгебра и логика, теория и приложения : тез. междунар. конф. Красноярск : СФУ, 2016. С. 22–23.
[3] Глушкова В. Н. Логические средства диагностики ошибок в иерархических S-моделях // Алгебра и математическая логика : мат-лы междунар. конф. Казань : КФУ, 2011. С. 58–59.
[4] Глушкова В. Н., Коровина К. С. Логическое моделирование роботизированных технологических систем // VIII Всерос. шк.-семинар : тез. Ростов н/Д : ЮФУ, 2013. С. 44.

списков $<t_1, \dots, t_n>$, формируемых для каждого правила $A \to X_1 \dots X_n \in P$, $n \geq 1$ следующим образом: t_i — произвольная константа из C_{X_i}, если $X_i \in T$; в противном случае для $X_i \in N$ элемент t_i — произвольный список сорта X_i.

Δ_0-формулы определяются традиционным способом как формулы сигнатуры σ с использованием всех логических связок ($\neg, \wedge, \vee, \to$) и ограниченных кванторов $\forall x \in t, \quad \exists x \in t, \quad \forall x \subseteq t, \quad \exists x \subseteq t$. Здесь x — переменная произвольного сорта; t — терм сорта *list*, не содержащий x; $\in$ — отношение принадлежности списку; $\subseteq$ — отношение вложенности для списков, определяемое как $<\alpha_1, \dots, \alpha_n> \subseteq <\alpha_1, \dots, \alpha_n>$, $m \geq n$.

Ниже будем использовать лишь ограниченные кванторы вида $\forall x \in y$, $\forall x \ \dot{\in}\ y$, где переменная y сорта *list*, отношение $\dot{\in}$ — транзитивное замыкание отношения $\in$. Обозначим через $\bar{x}$ индексированную последовательность переменных x_i, через $\mathring{\in}$ — отношение принадлежности или его транзитивное замыкание.

Правила КС-грамматики иерархизируют пространство действий моделируемой системы. Из соображений эффективности вычислений выделяется класс $\Delta_0 T$-формул с «древесным» префиксом. Введем для элементов списков отношение $\prec$ — «левее», а именно, для списка $< \cdots \alpha, \beta \dots >$ считаем $\alpha \prec \beta$.

Определение. Δ_0-формула вида

$$(\forall v_1 \mathring{\in} r_1) \dots (\forall v_m \mathring{\in} r_m) (n_1 \prec l_1) \dots (n_p \prec l_p)\ \Phi(\bar{v}, \bar{r}),\ m \geq 1, p \geq 0$$

называется $\Delta_0 T$-формулой, если $n_j, l_j \in (\bar{v}, \bar{r})$, $1 \leq j \leq p$; для всех переменных префикса выполняется условие: $r_{i+1} = r_i$, $1 \leq i < m$ или $r_{i+1} = v_k$, $k \leq i$. Если $r_{i+1} = v_k$, то $v_{i+k} \neq v_k$ и $v_{i+k} \neq r_k$ для всех $k \leq i$.

Легко показать, что префикс $\Delta_0 T$-формулы в силу ограничений на переменные можно представить в виде дерева с корнем r_1, вершинами v_i, r_i и дугами, идущими из вершины r_i в вершину v_i, $1 \leq i \leq m$.

Выразительных возможностей $\Delta_0 T$-формул недостаточно для моделирования систем реального времени, циклически функционирующих неограниченно долго. Будем использовать для спецификации ПТ-формулы с неограниченным квантором всеобщности $\forall$.

Определение. Формула, полученная из $\Delta_0 T$-формулы Φ навешиванием неограниченного квантора всеобщности $\forall v\ \Phi(v)$, называется ПТ-формулой.

Модель $\mathcal{M}$ определяется теорией из квазитождеств вида:

$$(\forall v_1 \dot{\in} r_1) \dots (\forall v_m \dot{\in} r_m)\ (n_1 \prec l_1) \dots (n_p \prec l_p)(\varphi(\bar{v}, \bar{r}) \to \psi(\bar{v}, \bar{r})).$$

Здесь формула φ (ψ) — это конъюнкция атомных формул (или их отрицаний) вида r, $\tau_1 = \tau_2$, $(f = \tau)$, $f \in F, r \in R$, τ_1, τ_2 — термы сигнатуры σ.

Алгоритм построения модели *Int* реализует правило вывода modus ponens (если φ и $\varphi \to \psi$, то ψ). Входные данные для интерпретатора — множество начальных значений функций и предикатов вида:

$$S_0 = \{p(\bar{c}), f(\bar{c}) = c_{n+1} | p \in P, f \in F\},$$

где $\bar{c}$ — множество констант из n элементов, $n \geq 1$.

Аксиомы (ax) обрабатываются в определенном порядке, сначала с позитивными вхождениями предикатов в левую и правую часть ax, потом с негативными вхождениями до получения неподвижных точек вычислений. Область определения функций и предикатов, входящих в правую часть ax, расширяется при истинности ее левой части. Это объясняется тем, что интерпретатор задает новые значения функций и предикатов так, чтобы правая часть ax была тоже истинной. Пусть состояние S_n анализируемой системы на -м шаге вычисления содержит значения всех предикатов и функций сигнатуры модели, а функция τ $P(S) \to P(S)$ в (терминологии [3] — преобразователь предикатов) отражает изменение состояния при переходе интерпретатора *Int* от n-го шага вычисления к n+1. Интерпретатор *Int* строит для монотонного преобразователя τ на $P(S)$ наименьшую неподвижную точку μZ.

$$\tau(Z) = \cup_i \tau^i(S_0), \text{ где } \tau^0(Z) = Z, \tau^{i+1}(Z) = \tau(\tau^i(Z)).$$

Формально функции $f \in F$ и предикаты $p \in R$ интерпретируются на КС-списке $tl(n)$, представляющем дерево вывода $tr(n)$ в грамматике G, где n — шаг работы *Int*. Из-за трудности представления компактной формы записи алгоритма интерпретации на элементах КС-списка приведем сначала словесное пояснение алгоритма, ориентируясь на дерево $tr(n)$. Входные данные для алгоритма построения модели (*Int*): исходное дерево вывода $tr(0)$ в грамматике G, расширяемое в процессе построения модели $\mathcal{M}$ и *Fact*=S_0. КС-грамматика используется в процессе построения модели следующим образом. Во-первых, правила P иерархизируют пространство действий и состояний анализируемой системы. Будем считать, что имена действий, представленные в сигнатуре модели предикатами, и имена соответствующих нетерминальных символов грамматики совпадают. Во-вторых, символы из алфавита V грамматики однозначно определяют сорта всех элементов универса модели,

включая списки, которым ставится в соответствие сорт, определяемый меткой корня соответствующего дерева. Сорта будем обозначать мнемонично начальными строчными символами имен нетерминальных и терминальных символов грамматики с добавлением в конце символа *s* (*sort*). Основное достоинство КС-грамматик состоит в возможности использования эффективных синтаксически ориентированных (СО) методов для анализа корректности (верификации) модели, развитых в теории синтаксического анализа языков программирования.

Интерпретатор начинает работу с просмотра дерева $tr(0)$ от корня сверху вниз, слева направо. Префикс всех аксиом удовлетворяет ограничениям $\Delta_0 T$-формул. Сорта префикса определяются символами КС-грамматики G. Интерпретатор выбирает в качестве констант области истинности предикатов, входящих в аксиому $ax \in Th$, константы, сопоставляемые вершинам куста дерева, просматривая его сверху вниз, слева направо. Причем корень куста помечен именем соответствующего предиката. Для отражения зависимости моделируемой технической системы от последовательности входных сигналов необходимо достраивать исходное дерево $tr(0)$. С этой целью последовательность правил $pr^{+}(ax) \in P^{+}$ приписывается к выводу дерева, полученного на предыдущем шаге работы алгоритма. Причем константы из области истинности предиката r используются в качестве терминальных символов, подчиненных в дереве вершине, помеченной нетерминальным символом r.

Опишем алгоритм интерпретации Int более формально, не детализируя процедуру $Con(Q, Th)$ — получения всех логических следствий из множества формул Q на основе аксиом теории Th. $Th_0 = S_0$. Теория $Th_{pos} \subseteq Th$ включает только позитивные вхождения предикатов. $Th_{neg} \subseteq Th$ включает негативное вхождение предикатов в правой части аксиом. Обозначим tr_{pos}, tr_{neg} — деревья выводов, генерируемые в процессе интерпретации.

Q:=∅;
Q':=Th_0;
while $Q \neq Q'$ do
$Q_{pos} := Q$;
$Q'_{pos} := Q'$;
while $Q_{pos} \neq Q'_{pos}$ do
$Q_{pos} := Q'_{pos}$;

$$Q'_{pos} := Con(Q'_{pos}, Th_{pos})$$

end while
return (Q_{pos} , tr_{pos}) ;
$Q_{neg} :=\ \emptyset$;
$Q'_{neg} := Q_{pos}$;
while $Q_{neg} \neq Q'_{neg}$ do
$Q_{neg} := Q'_{neg}$;

$$Q'_{neg} :=\ Con(Q'_{neg}, Th_{neg})$$

end while
return (Q_{neg}, tr_{neg});

$$Q := Q_{pos};$$
$$Q' := Q_{neg}$$

end while
return (Q_{neg}, tr_{neg})

Верификация модели $\mathcal{M}$ состоит в проверке свойств, которым должна удовлетворять анализируемая система. Будем выражать эти свойства произвольными $\Delta_0 T$-формулами. Используя СО-методы проверки формул, можно строить доказательство аналогично [13].

Теорема. Произвольная $\Delta_0 T$-формула с *m*-ограниченными кванторами всеобщности проверяется на КС-списке мощности n за время $O(n^{m+1})$.
Мощность списка tl равна мощности множества $\{s \mid s \dot{\in} tl\}$.

Оценка является верхней, и ее можно понизить до линейной, если проверять формулы с использованием специфичных СО-методов обработки языков.

Результаты исследования. Приведем логическую спецификацию автоматизированного комплекса, состоящего из манипулятора, обслуживающего технологическую линию (*tl*) с двумя позициями: загрузки и разгрузки деталей (*ld* и *uld* соответственно) [14]. Датчики фиксируют положение позиций. Манипулятор функционирует циклически, начиная с позиции загрузки.

ЦИКЛ

1. В начальном положении *ld* для загрузки детали манипулятор поднимает электропривод за 4 сек. Сжимает автоматизированные клещи и берет заготовку (2 сек), опускает электропривод (4 сек) и передвигается вправо к автомату до срабатывания датчика положения *tl*.

2. Чтобы установить заготовку на автомате на позиции *tl*, манипулятор поднимает электропривод, разжимает автоматизированные клещи (2 сек), опускает электропривод. Далее манипулятор ожидает 4 мин, после чего повторяет те же процедуры, что и на позиции *ld*. Затем манипулятор двигается влево к позиции разгрузки до срабатывания концевого выключателя *uld*.

3. За 8 сек деталь разгружается на конвейер. Манипулятор передвигается влево до фиксации датчика *ld* на позицию загрузки. Далее процесс работы комплекса циклически повторяется.

Спецификация системы состоит из нескольких уровней. Поведение манипулятора определяется сигналами датчиков, фиксирующих его положение: *ld*, *uld*, *tl* (¬*ld*, ¬*uld*, ¬*tl*, отрицание указывает на отсутствие соответствующего сигнала). Эта последовательность сигналов представлена кортежем *mc*=<*x*, *y*, *z*...>, где *x* = *ld* (¬*ld*) , *y* = *uld* (¬ *uld*), *z*=*tl* (¬ *tl*). Она генерируется конечным автоматом с начальным состоянием *x*. Этот автомат строится по праволинейной грамматике с правилами:

$St \rightarrow ld\ St_1 \mid \neg\ ldSt_1 \mid \varepsilon$

$St_1 \rightarrow tl\ St_2 \mid \neg\ tl\ St_2$

$St_2 \rightarrow uld\ St \mid \neg\ uld\ St.$

Обозначим через *Dsig* множество списков, составленных из цепочек символов, порожденных этой грамматикой.

Внешнее дискретное время (переменная *n* в логической спецификации) определяется количеством переходов в автомате. Для описания второго уровня функционирования манипулятора используется КС-грамматика, которая указывает последовательность действий (*Oper*) манипулятора:

— *L*, *La* — загрузка заготовки манипулятором в положении *ld* и *tl* соответственно;

— *Unl*, *Unla* — разгрузка детали в позиции *uld* и *tl*;

— *Mover*, *Movel* — движение манипулятора направо и налево;

— *Lstop*, *Astop*, *Ulstop* — остановка манипулятора в соответствующем положении;

— *Exp* — ожидание;

— *Cr* — выход из строя управляющего манипулятором устройства.

На состояния манипулятора (символ *Pos*) влияют его действия. В указанном примере состояние характеризуется непрерывным временем *Timec* и дискретным *Timed*, задаваемым натуральным числом. Значением сорта *Timec* являются сегменты вида $< t_1,\ t_2 >,\ t_1,\ t_2$ — константы, причем < заменяется на (или [в зависимости от того, включена левая граница в сегмент времени или нет, аналогично для >.

При спецификации поведения манипулятора абстрагируются от значения времени перемещения манипулятора из одного положения в другое (определяется сигналами с датчиков положения — входными к устройству управления манипулятора). Сигналы, которые подаются на исполнительные механизмы манипулятора для осуществления передвижений и работы автоматизированных клещей, являются выходными.

Ниже представлены правила грамматики *G*.

1. $Start \rightarrow \{Oper\}^*.$
2. $Oper \rightarrow L \mid La \mid Unl \mid Unla \mid Mover \mid Movel \mid Lstop \mid Astop \mid Ulstop \mid Exp \mid Cr.$
3. $L \rightarrow St.$
4. $La \rightarrow St.$
5. $Unl \rightarrow State.$
6. $Unla \rightarrow State.$
7. $Mover \rightarrow State.$
8. $Movel \rightarrow State.$
9. $Lstop \rightarrow State.$
10. $Astop \rightarrow State.$
11. $Ulstop \rightarrow State.$
12. $Exp \rightarrow State.$
13. $Cr \rightarrow State.$
14. $State \rightarrow Timec\ Timed \mid Timed\ Timed.$
15. $Timec \rightarrow Timed \mid (Timed, Timed) \mid [Timed, Timed) \mid (Timed, Timed] \mid [Timed, Timed].$

Timed — класс лексем, значениями которых являются натуральные числа, вычисляемые во время интерпретации теории *Th.*

В теории *Th* переменные в формулах обозначаются мнемонично в соответствии с их сортом: ρ (*state*) = ρ (*State*), ρ (*oper*) = ρ (*Oper*), ρ (*n*) = ρ (*t*) = ρ (*Timed*), ρ (*ct*) = ρ (*Timec*). Предикаты *Ld*, *Tl*, *Unld* определены на множестве *Timed*. *Ld* (*n*) истинен, если манипулятор находится на позиции загрузки. Аналогично для *Tl* (*n*) — на позиции обрабатывающего автомата, *Uld* (*n*) — на позиции разгрузки. Перечислим области определения остальных предикатов: *Lstop*, *Astop*, *Ulstop*, *Mover*, *Movel*, *Cr* ⊆ *Timed*×*Timed*; *L*, *La*, *Unl*, *Unla*, *Exp*⊆*Timec* × *Timed*. В формулах используются стандартные функции *head* $(<x_1,...x_n>) = x_1$, *tail*$(<x_1,...x_n>) = <x_2,...x_n>$ и функция *Mc*: *Timed*→ *Dc*. Здесь *Dc* — множество списков из сигналов датчиков.

В начальный момент времени *t* = 0, *n* = 1 и на 1-м шаге вычисления выполняется предикат *Lstop*(0,1); *Mc* (1) = *mc*, где *mc*∈*Dc*. В аксиомах 1–11 переменные *t*, *n* и *ct* связаны ограниченным квантором ∀ *state* $\overset{*}{\in}$ *oper*, ∀ *t*,*n*, *ct* ∈ *state*. В аксиомах 12–17 переменная *n* связана неограниченным квантором ∀; s_0 = <<<<0,1>>>> — начальное значение списка, на котором интерпретируется теория. Составляющие его списки в порядке глубины вложенности имеют сорта ρ (*R*), ρ (*Oper*), ρ (*Lstop*), ρ (0) = ρ (1) = ρ (*Timed*). Списку s_0 соответствует дерево T_0 (рис. 1).

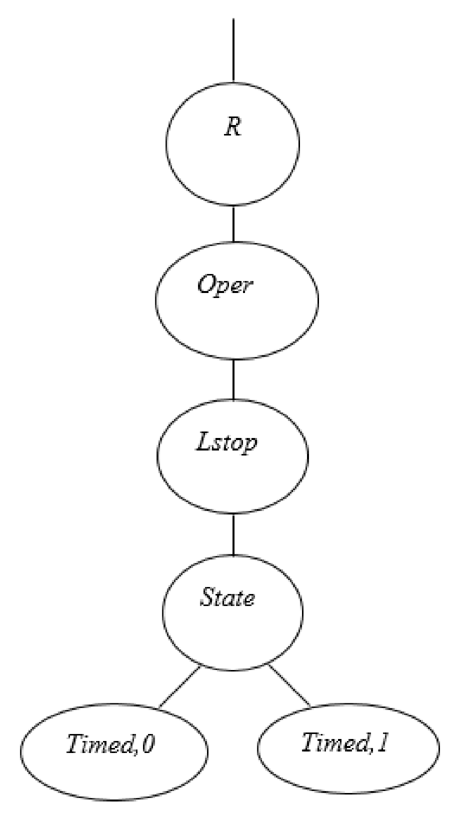


Рис. 1. Дерево T_0, соответствующее списку s_0

В аксиомах теории справа в квадратных скобках приводится последовательность правил грамматики *G*, достраивающих дерево T_0.

Аксиомы теории

1. *Lstop*(*t*, *n*) ∧ *Ld* (*n*) → *L*([*t*, *t* + 7), *n*) ∧ *Mc* (*n* + 1) = *tail*(*Mc* (*n*)) [1; 2.1; 3; 14.1; 15.3].
2. *L*(*ct*, *n*) ∧ *Tl*(*n* + 1) → *Mover* (*ct* [2], *n* + 1) ∧ *Astop* (*ct* [2], *n* + 1) [1; 2.5; 7; 14.2;1; 2.8; 10; 14.2].
3. *Astop* (*t*, *n*)→ *Unla* ([*t*, *t* + 7], *n*) [1; 2.4; 6; 14.1; 15.2].
4. *Unla* (*ct*, *n*) → *Exp* ((*ct* [2], *ct* [2] + 180), *n*) [1; 2.10; 11; 14.1; 15.2]
5. *Exp* (*ct*, *n*) → *La* ([*ct* [2], *ct* [2] + 3), *n*) ∧ *Mc* (*n* + 1) = *tail*(*Mc* (*n*)) [1; 2.2; 4; 14.1; 15.2].
6. *La* (*ct*, *n*) ∧ *Uld* (*n* + 1) → *Movel* (*ct* [2], *n* + 1) ∧ *Ulstop* (*ct* [2], *n* + 1) [1; 2.6; 8; 14.2; 1; 2.9; 11; 14.2].
7. *Ulstop* (*t*, *n*) → *Unl* ([*t*, *t* + 7), *n*) ∧ *Mc* (*n* + 1) = *tail*(*Mc* (*n*)) [1; 2.3; 5; 14.1; 15.2].
8. *Unl* (*ct*, *n*) ∧ *Ld* (*n* + 1) → *Movel* (*ct* [2], *n* + 1) ∧ *Lstop* (*ct* [2], *n* + 1) [1; 2.6; 8; 14.2; 1; 2.7; 9; 14.2].
9. *Unl* (*ct*, *n*) ∧ ¬ *Ld* (*n* + 1) → *Cr* (*ct* [2], *n* + 1) [1; 2.11; 13; 14.2].

10. $La\ (ct, n) \wedge \neg\ Unld\ (n + 1) \rightarrow Cr\ (ct\ [2], n + 1)$ [1; 2.11; 13; 14.2].
11. $L\ (ct, n) \wedge \neg\ Tl\ (n+1) \rightarrow Cr\ (ct\ [2], n + 1)$ [1; 2.11; 13; 14.2].
12. $head\ (Mc\ (n)) = ld \rightarrow Ld\ (n)$.
13. $head\ (Mc\ (n)) = \neg\ ld \rightarrow \neg\ Ld\ (n)$.
14. $head\ (Mc\ (n)) = tl \rightarrow Tl\ (n)$.
15. $head\ (Mc\ (n)) = \neg tl \rightarrow \neg\ Tl\ (n)$.
16. $head\ (Mc\ (n)) = uld \rightarrow Unld\ (n)$
17. $head\ (Mc\ (n)) = \neg uld \rightarrow \neg\ Uld\ (n)$.

Теория *Th* обладает свойством нётеровости, т. к. изменение переменной под квантором ∀ ограничено *k* — количеством элементов в исходном списке *mc*. При этом *head* (*Mc* (*k*+1)) не определено, т. к. *Mc* (*k*+1)=< >. Отметим, что цепочки ¬*ld* и др. в правой части аксиом 12–17 имеют сорт *string*, и ¬ рассматривается не как логическая операция, а как символ.

Для начального значения функции *Mc* (1) = <*ld*, *tl*, *uld*, *ld*, ¬ *tl*, *uld* > получаем множество следствий: *Lstop*(0,1), *Ld* (1), *L* ([0,7), 1), *Mc* (2) = < *tl*, *uld*, *ld*, ¬ *tl*, *uld*>, *Mover* (7,2), *Astop* (7,2), *Unla* ([7, 14], 2), *Exp*([14, 194], 2), *La*([194, 197), 2), *Mc* (3) = <*uld*, *ld*, ¬ *tl*, *uld*>, *Movel* (197, 3), *Ulstop* (197, 3), *Unl* ([197, 204], 3), *Mc* (4) = <*ld*, ¬ *tl*, *uld* >, *Movel* (204, 4), *Lstop* (204, 4), *L* ([204, 211), 4), *Mc* (5) = < ¬ *tl*, *uld*>, *Cr* (211, 5).

Полученное множество следствий иерархизируется в соответствии с выводом в грамматике *G*, полученным в результате правил, приписанных к интерпретируемым аксиомам. В соответствии с ними к дереву T_0 справа от узла, помеченного символом *Oper*, добавляются еще 12 вершин, помеченных этим же символом и связанных ребрами с корнем. К новым вершинам подвешиваются поддеревья с корнями, помеченными символами *Ld*, *Mover*, *Astop* и т. д. с их состояниями и константами сорта ρ (*Timed*), полученными в результате интерпретации.

Обсуждения и заключения. На построенной модели можно проверять истинность произвольных $\Delta_0 T$-формул. Например, формализуем утверждение: «Если манипулятор стоял на позиции загрузки в момент *t* на шаге *n* цикла его функционирования, то на шаге *n* + 2 через 197 сек он начинает разгрузку в течение 7 сек». Приведенная ниже формула проверяется на заданном списке *Oper* сорта ρ (*Oper*):

$$(\forall state \overset{*}{\in} oper)\ (\forall t \in State)\ (\forall n \in state)\ (Lstop(t, n) \rightarrow Unl\ ([t + 197, t + 204], n + 2).$$

Библиографический список

1. Goguen, J. A. Models and equality for logical programming / J. A. Goguen, J. Meseguer // Lecture Notes in Computer Science. — 1987. — Vol. 250. — P. 1–22.

2. Kowalski, R. Logic for Problem Solving, Revisited / R. Kowalski. London : Imperial College, 2014. — P. 321.

3. Кларк, Э. М. Верификация моделей программ / Э. М. Кларк, О. Грамберг мл., Д. Пелед // Москва : Изд-во Московского центра непрерывного математического образования, 2002. — 416 с.

4. Reps, T. Automating Abstract Interpretation / T. Reps, A. Thakur // In: 17th International Conference, VMCAI 2016, on Verification, Model Checking and Abstract Interpretation. St. Petersburg, FL, USA, January 17–19, 2016. — Paris : Springer, 2016. — P. 3–40.

5. Bloem, R. SAT-Based Synthesis Methods for Safety Specs / R. Bloem, R. Konighofer, M. Seidl // In: 15th International Conference, VMCAI 2014, on Verification, Model Checking and Abstract Interpretation. San Diego, CA, USA, January 2014. — San Diego : Springer, 2014. — P. 1–20.

6. Beyer, D. Reuse of Verification Results / D. Beyer, Ph. Wendler // In: 20th International Symposium, SPIN 2013, on Model Checking Software. Stony Brook, July 8–9, 2013. — Stony Brook : Springer, 2013. — P. 1–15.

7. Alur, R. Model-cheking for real-time system / R. Alur, C. Courcoubetis, D. L. Dill // Information and Computation. — 1993. — Vol. 104 (1). — P. 2–34.

8. Morbe, G. Fully Symbolic TCTL Model Checking for Incomplete Timed Systems // G. Morbe, Ch. Scholl // In: Proceedings of the Automated Verification of Critical Systems (AVoCS 2013). — 2013. — Vol. 66. — P. 1–9.

9. D´Silva, V. Independence Abstractions and Models of Concurrency / V. D´Silva, D. Kroening, M. Sousa // In: 18th International Conference, VMCAI 2017, on Verification, Model Checking and Abstract Interpretation. Paris, France, January 15–17, 2017. — Paris : Springer, 2017. — P. 149–168.

10. Goncharov, S. S. Theoretical aspects of Σ-programming / S. S. Goncharov, D. I. Sviridenko //: In: Proc. of the International Spring School, April 1985, on Mathematical Methods of Specification and Synthesis of Software Systems' 85. Berlin ; Heidelberg : Springer-Verlag, 1985. — P. 169–179.

11. Гончаров, С. С. Модели данных и языки их описаний / С. С. Гончаров // Вычислительные системы. Логико-математические основы проблемы МОЗ. — 1985. — Вып. 107. — С. 52–70.

12. Мальцев, А. И. Алгебраические системы. Москва : Наука, 1976. — С. 392.

13. Глушкова, В. Н. Оценка сложности реализации логических спецификаций на основе контекстно-свободных грамматик / В. Н. Глушкова // Кибернетика и системный анализ. — 1996. — № 4. — С. 50–58.

14. Горбатов, В. А. Логическое управление распределенными системами / В. А. Горбатов, М. И. Смирнов, И. С. Хлытчиев. — Москва : Энергоатомиздат, 1991. — 288 с.

Сдана в редакцию 27.07.2020
Запланирована в номер 05.10.2020

Об авторах:

Глушкова Валентина Николаевна, доцент кафедры «Математика» ФГБОУ ВО Донской государственный технический университет (344003, РФ, г. Ростов-на-Дону, пл. Гагарина, 1), кандидат физико-математических наук, старший научный сотрудник, ORCID: https://orcid.org/0000-0003-2719-8590, lar@aaanet.ru.

Коровина Ксения Сергеевна, старший преподаватель кафедры «Математика» ФГБОУ ВО Донской государственный технический университет (344003, РФ, г. Ростов-на-Дону, пл. Гагарина, 1), ORCID: https://orcid.org/0000-0002-1196-3596, ksenichka_@inbox.ru.

Заявленный вклад соавторов:

В. Н. Глушкова — формирование концепции синтаксически ориентированного иерархического моделирования, цели и задачи логической спецификации, научное руководство, корректировка выводов исследования. К. С. Коровина — применение Σ-формул для спецификации технической системы, анализ результатов верификации системы и формирование выводов, доработка текста.

Все авторы прочитали и одобрили окончательный вариант рукописи.